Chapter 1

Introduction to Cloud Computing

1.1 Introduction

In the contemporary era, where ubiquitous access to technology is a paramount expectation, Cloud Computing emerges as a transformative solution, offering the flexibility of accessing resources anytime and anywhere. Despite the abundance of resources, their efficient utilization has remained a challenge, even with the advent of distributed computing. Cloud Computing addresses this challenge by providing a model that enables ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources. This model, defined by the National Institute of Standards and Technology(NIST), allows for the rapid provisioning and release of resources with minimal management effort or service provider interaction [54].

The representation of the "Internet" in network architecture diagrams as shown in Fig. 1.1 often features a cloud symbol, leading to the term "Cloud Computing." This paradigm shift has revolutionized computing, allowing users to access computing resources through a cloud provided by servers. The underlying principle revolves around delivering everything "as a Service" [54]. Various online vendors, including Amazon, Google, and Microsoft, provide on-demand computing facilities, contributing to the widespread adoption of Cloud Computing [79].

1.1.1 Cloud Service Models

Cloud computing services are categorized into Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) [54]. IaaS provides virtualized resources such as on-demand storage, offering users flexibility in resource allocation [64]. PaaS elevates the level of abstraction, enabling users to program applications without intricate knowledge of underlying hardware requirements. It provides a platform for

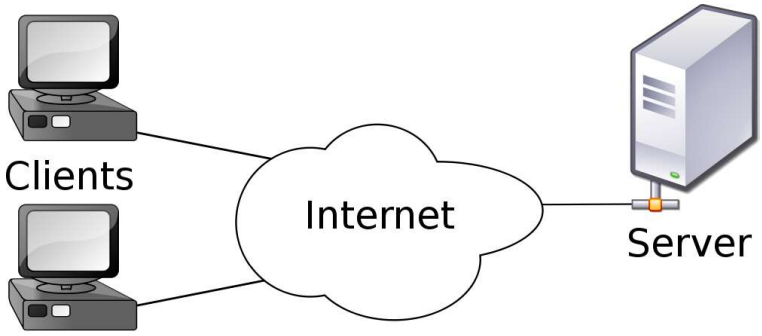

Fig. 1.1 Model of client- server architecture

application development and deployment. SaaS, on the other hand, grants users access to software applications over the internet, relieving them from the responsibilities of software maintenance.

The cloud computing stack encompasses these service models, creating a hierarchical arrangement that reflects the provisioning of infrastructure, platforms, and software applications (see Fig. 1.2).

Infrastructure as a Service (IaaS)

IaaS serves as the foundational layer of cloud services, providing virtualized computing resources over the internet. Users have control over the operating system, applications, and development frameworks. Resources are scalable, and users can provision and manage virtual machines as needed. However, users are responsible for managing the operating system, applications, and data.

Platform as a Service (PaaS)

PaaS offers a higher-level service that provides a platform allowing customers to develop, run, and manage applications without dealing with the complexities of infrastructure. It includes development tools, databases, middleware, and other services necessary for application development and deployment. Users can focus on building and deploying applications without managing the underlying infrastructure, as platform updates and maintenance are handled by the service provider.

Software as a Service (SaaS)

SaaS represents the top layer of cloud services, delivering software applications over the internet on a subscription basis. Users can access software applications through a web browser without needing to install or maintain the software locally. Maintenance, updates,

and security are handled by the service provider, and users are typically charged on a subscription or pay-per-use basis.

Advantages of Cloud Service Models

Cloud service models offer several advantages, including scalability, cost efficiency, resource management, accessibility, reliability, and automatic updates. These benefits make cloud computing an attractive option for organizations seeking flexible and efficient IT solutions.

Considerations and Challenges

However, organizations considering the adoption of cloud service models must address certain considerations and challenges. These include security concerns, data privacy, customization limitations, integration challenges, and dependency on service providers.

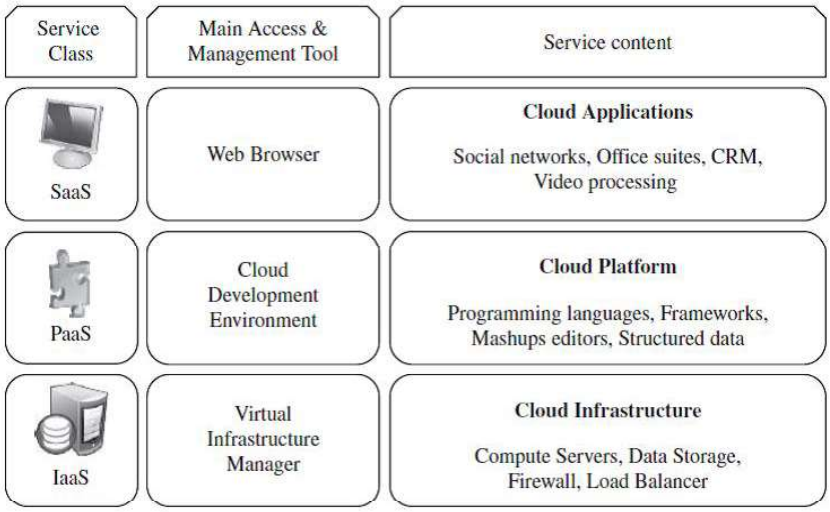

Fig. 1.2 Cloud computing service models

1.1.2 Deployment Models

In a cloud computing environment designed for deployment purposes, four typical types of clouds are recognized. The classification includes:

- **Public Cloud:** A public cloud is a cloud infrastructure available to users as a pay-per-use, on-demand service accessible to the general public. In this, the services are provided by third-party cloud service providers and resources are shared among multiple users on the public cloud platform. Users pay for the resources they consume, and the infrastructure is hosted off-site.

- **Private Cloud:** A private cloud involves modifying an existing service by incorporating the concept of virtualization, making it exclusively accessible to a particular user or organization. In this, resources are dedicated to a single user or organization. The cloud infrastructure may be hosted on-premises or by a third-party service provider. It offers enhanced control and customization over the cloud environment.

- **Community Cloud:** A community cloud is shared among various communities, such as individuals or groups of individuals who share common concerns. The resources are shared by multiple organizations or entities with common interests or requirements. It facilitates collaboration and resource sharing within a specific community. It also offers a balance between public and private cloud characteristics.

- **Hybrid Cloud:** A hybrid cloud is an environment where the properties of both private and public clouds are combined. It involves the integration of on-premises infrastructure (private cloud) with public cloud services. It offers flexibility and scalability by allowing data and applications to move between private and public environments. It enables organizations to leverage the advantages of both cloud deployment models.

These cloud deployment models provide organizations with flexibility and options based on their specific needs, security requirements, and resource utilization preferences [64]. The choice of the deployment model depends on factors such as data sensitivity, control requirements, and the need for collaboration with other organizations.

1.2 Scheduling in Cloud Computing

Cloud computing has emerged as a transformative paradigm, providing ubiquitous access to computing resources. However, the efficient utilization of these resources poses a significant challenge, necessitating the development of effective scheduling mechanisms. Scheduling in cloud computing involves the allocation of computational tasks to available resources, with the overarching goal of maximizing resource utilization, minimizing task completion time, and enhancing overall system performance. Several criteria guide the optimization of a schedule in cloud computing, each addressing specific aspects of performance. The key criteria for schedule optimization are as follows:

- **Execution Cost:** Cloud service is not free. It is on demand pay per use model, so a user has to pay some amount to access the resources offered by cloud service providers. The optimizing criteria for cloud task scheduling can be in such a way that it reduces the total execution cost of running tasks on processors.

- **Makespan:** Makespan is the time at which the last task finishes its execution. Another criteria can be said where the optimisation is done on the basis of makespan and the schedule with the least makespan is considered to be the most optimized one.

- **Waiting Time:** It can be said the time between the execution start time and the submission time of the task. Reducing the waiting time can be considered to be an optimizing criteria for cloud task scheduling.

- **Deadline:** Deadline can= be defined as the time before which a task should complete it's execution and the most optimized schedule will be when most number of tasks finish before their given deadline.

- **Completion time:** It is the total time of all the tasks to be completed. The optimized mapping is said to be having the schedule that has least completion time for given tasks and processors.

These optimization criteria play a crucial role in determining the efficiency and effectiveness of cloud task scheduling. Depending on the specific requirements and priorities of users, scheduling algorithms can be tailored to minimize costs, enhance task completion speed, and meet time-sensitive constraints. The choice of optimization criteria depends on the overarching goals and constraints of the cloud computing environment. Two fundamental types of scheduling in cloud computing are Task Scheduling and Workflow Scheduling.Let's explore each of these scheduling types in the upcoming sections.

1.2.1 Independent Task Scheduling

Independent task scheduling in a cloud environment refers to the allocation of computational resources for a set of tasks that do not have dependencies on each other. These tasks can be executed concurrently, allowing for parallel processing and efficient resource utilization. The scheduling process involves assigning each independent task to an available processing unit, such as a virtual machine (VM) in a cloud infrastructure. The task scheduling problem in cloud computing, as depicted in Fig. 1.3, is a complex optimization challenge given the multitude of user requests and the need to efficiently allocate resources in a large-scale cloud infrastructure. The goal is to provide an optimal schedule for running tasks on virtual machines, taking into account all the factors such as computation power, communication cost, and task size, etc. Due to the NP-hard nature of this problem, various optimization algorithms are employed to devise optimal schedules.

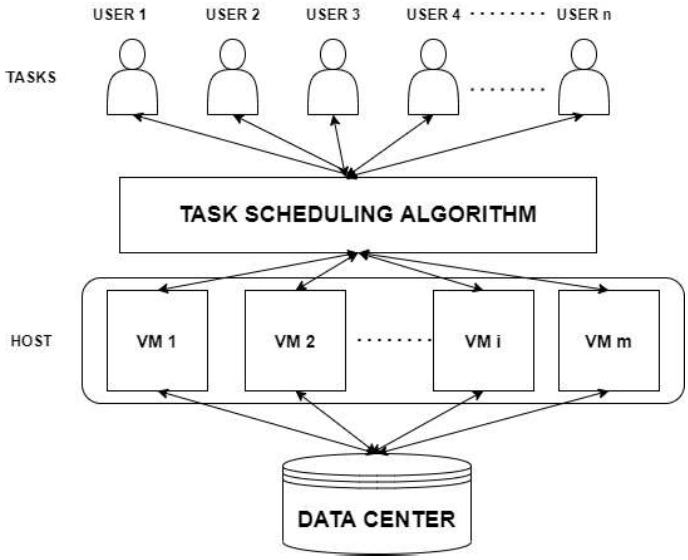

Fig. 1.3 Scheduling of tasks in cloud computing

Components of the Scheduling System:

- **Users and Requests:**
  1. Millions of users generate requests to utilize cloud resources.
  2. Each request represents a task to be processed in the cloud.
- **Virtual Machines (VMs) and Hosts:**
  1. VMs, hosted on different hosts within the **DataCentre**, represent processors with varying computation powers.
  2. VMs are available to execute tasks requested by users.
- **DataCentreBroker:**
  1. Maintains a list of tasks generated by users and available VMs in the cloud.
  2. Acts as an intermediary between users and the cloud infrastructure.
- **VMScheduler:**
  1. Responsible for assigning tasks to VMs based on optimization algorithms.
  2. Aims to minimize the total execution cost by judiciously mapping tasks to appropriate VMs.

Task scheduling involves assigning a collection of tasks to a set of processors, where processors can be physical machines or virtualized computing units. The objective is to create an optimal mapping that minimizes the overall execution cost. The mapping is optimized based on various criteria, including execution time, resource utilization, and cost-effectiveness. The Fig. 1.4 illustrates an example mapping where tasks (Task1, Task2, etc.) are allocated to specific processors (P1, P2, etc.).Each task in the set has specific computational requirements, execution time, and resource dependencies. The mapping aims to balance the load across processors and efficiently utilize available resources. Processors can be physical machines or virtual machines in a cloud environment. The allocation is done based on the characteristics of both tasks and processors. The primary objective is to minimize the overall execution cost, which may include factors like computation time, energy consumption, and resource usage. In dynamic environments, the mapping may change over time based on the evolving nature of tasks and available resources. Effective mapping involves load balancing, ensuring that no processor is heavily loaded while others remain underutilized. Load balancing contributes to optimal resource usage and improved performance.Dynamic mapping allows for adaptability to variations in workloads.

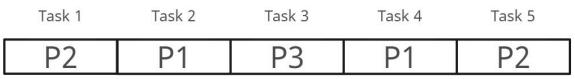

Fig. 1.4 A sample mapping or schedule showing the allocation of processors to tasks

1.2.2 Workflow Scheduling in Cloud

Workflow

In a workflow schedule, the tasks are dependent on each other. For eg. in Fig. 1.5, the nodes represent the tasks which are waiting to be allotted to processors. The edges represent the dependency. The height of the tree is 3, so there are 3 levels. The root which is in level 1, is always the first task waiting to be allotted to a processor. The tasks in level 2 can be allotted to processors until all the tasks in level 1 have finished their execution. Thus, in a workflow schedule, a level of tasks can be executed only and until the tasks in the previous level have been finished. Scientific benchmark workflows play a crucial role in assessing and comparing the performance of computational systems, algorithms, and infrastructures. Several scientific benchmarks are designed to represent the computational workloads of specific scientific applications. Here, we'll explore four notable scientific benchmark workflows: Montage, SIPHT, Inspiral, and CyberShake which can be seen in Fig. 1.6.

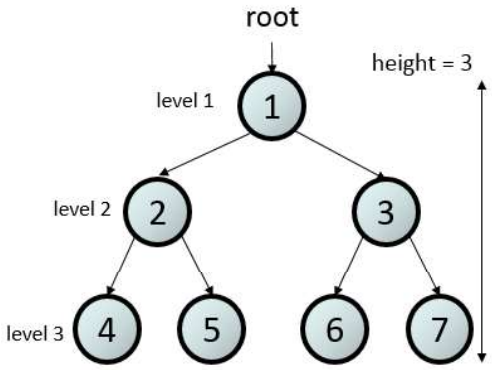

Fig. 1.5 Dependency of tasks for workflow scheduling

Description of Various Benchmark Workflows

i **Montage**:

- Overview:
  Montage, tailored for astronomical image mosaic processing, is instrumental in creating comprehensive sky surveys. This benchmark addresses the challenges of handling vast amounts of astronomical data and processing images efficiently.
- Characteristics:
  - Tasks: Image reprojection, background matching, and co-addition. Applications: Cloud infrastructure and high-performance computing (HPC) systems in large-scale astronomical data processing.
  - Significance: Montage enables researchers to assess the efficiency of distributed computing systems and HPC environments in managing complex tasks associated with astronomical image processing.

ii **SIPHT**:

- Overview:
  SIPHT, or Scalable Integrated Performance environment for HPC Tasks, is specifically crafted for fusion research, focusing on gyrokinetic continuum simulations. It addresses the computational challenges inherent in simulating complex plasma physics phenomena.
- Characteristics:

  - Tasks: Gyrokinetic continuum simulations, particle distribution functions, and result analysis.
  - Applications: Evaluating HPC systems for large-scale fusion simulations.
  - Significance: SIPHT allows for an in-depth examination of the scalability and efficiency of HPC systems in capturing intricate fusion research simulations, providing insights into optimal configurations.

iii **Inspiral (LIGO)**:

- Overview:
  Inspiral, associated with the Laser Interferometer Gravitational-Wave Observatory (LIGO), focuses on the detection and characterization of gravitational waves resulting from compact binary coalescence events.
- Characteristics:
  - Tasks: Gravitational wave signal identification, analysis, and characterization.
  - Applications: Assessing computing systems for gravitational wave data analysis.
  - Significance: Inspiral plays a vital role in evaluating the capability of computing systems to analyze vast datasets generated by gravitational wave detectors, contributing to advancements in astrophysics.

iv **CyberShake**:

- Overview:
  CyberShake is a seismic hazard analysis benchmark workflow designed for simulating ground motions and earthquake scenarios. It addresses the complexities of seismic event simulations and their implications for infrastructure resilience.
- Characteristics:
  - Tasks: Seismic hazard analysis, ground motion simulations.
  - Applications: Assessing high-performance computing systems for seismic risk analysis.
  - Significance: CyberShake aids in evaluating the suitability of computing systems for seismic hazard analysis, offering valuable insights into optimizing simulations for diverse earthquake scenarios.

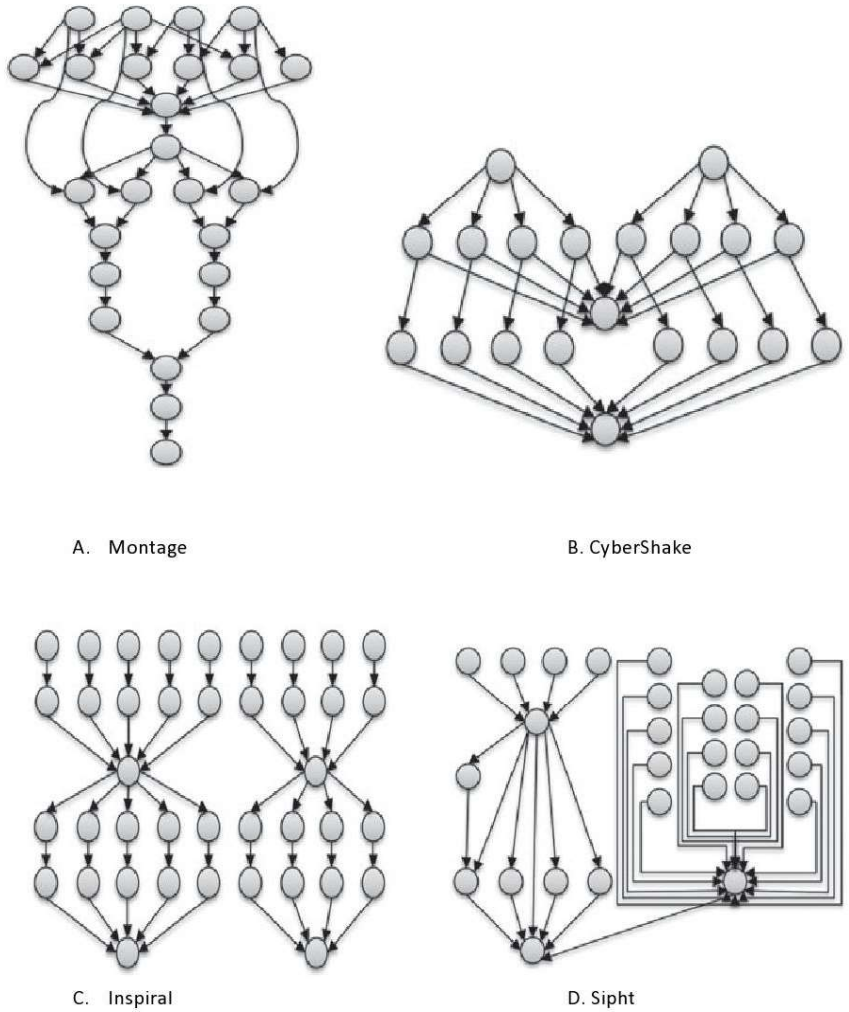

Fig. 1.6 Workflows used

Workflow Scheduling

Workflow scheduling in cloud computing refers to the process of efficiently allocating and executing a set of interdependent tasks within a cloud environment. This scheduling is crucial for optimizing resource utilization, meeting performance requirements, and ensuring timely completion of tasks in complex workflows. In the context of cloud computing, where resources are shared and dynamically provisioned, effective workflow scheduling becomes a key challenge.
Key Considerations in Workflow Scheduling in Cloud Computing:

i **Task Dependencies:** Workflows consist of multiple tasks that often have dependencies on each other. Efficient scheduling requires considering these dependencies to ensure that tasks are executed in the correct order.

ii **Resource Management:** Cloud environments offer a pool of virtualized resources, including virtual machines (VMs) with varying capabilities. Workflow scheduling involves mapping tasks to suitable VMs to optimize resource utilization and meet performance objectives.

iii **Dynamic Nature of Cloud:** Cloud environments are dynamic, with varying workloads and resource availability. Workflow schedulers must adapt to changes in real-time, making dynamic scheduling strategies essential.

iv **Cost Considerations:** Cloud services often come with costs, and efficient scheduling involves minimizing these costs while meeting workflow deadlines. This may involve considerations of resource pricing, data transfer costs, and other relevant financial factors.

v **Fault Tolerance:** Cloud environments may experience failures or disruptions. Workflow schedulers should incorporate mechanisms for fault tolerance to ensure the reliability of the overall system.

vi **Performance Metrics:** Different workflows may have distinct performance metrics, such as makespan (total time to complete the workflow), throughput, and response time. Scheduling strategies need to be tailored to meet the specific requirements of the workflow.

Process of Workflow Scheduling

Workflow scheduling in the cloud involves efficiently allocating and managing computational resources to execute a series of tasks constituting a workflow. The goal is to optimize

resource utilization, minimize execution time, and possibly reduce costs. The process of workflow scheduling can be well understood by Fig. 1.7 as depicted in [6]. The process

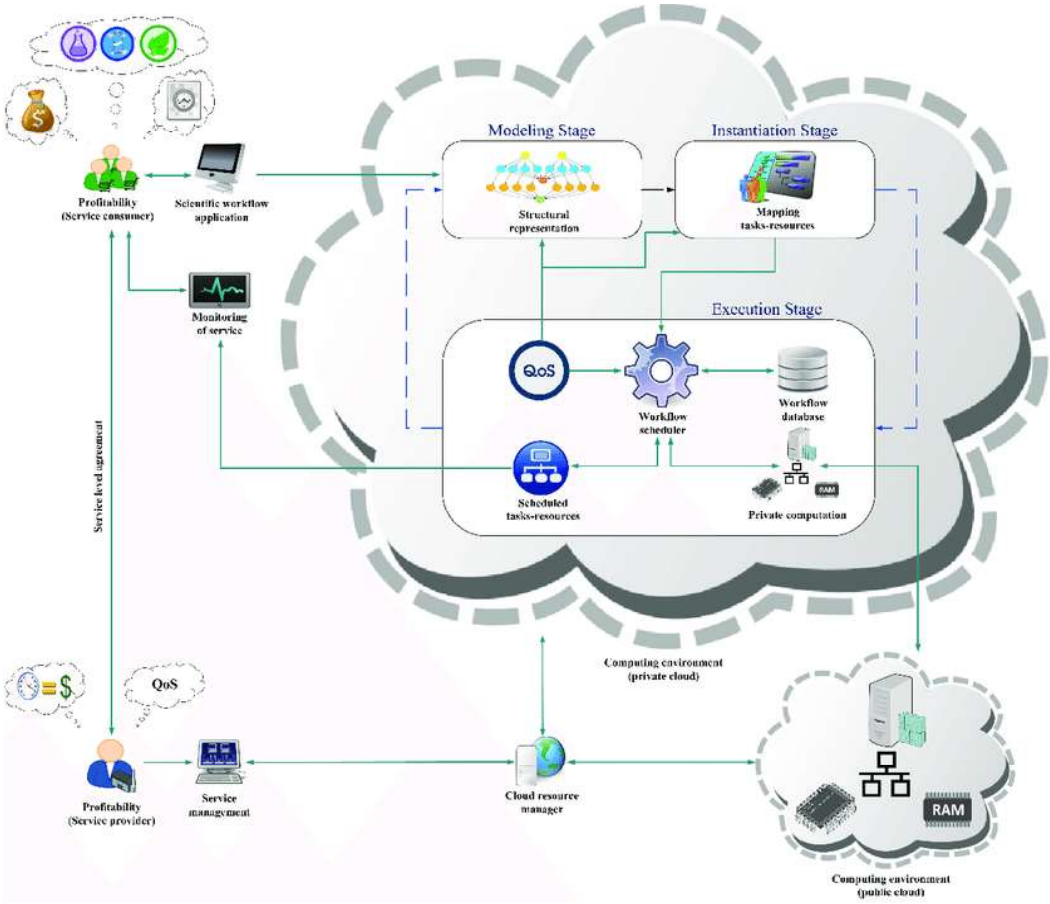

Fig. 1.7 Process flow for workflow scheduling in cloud[6]

can be broken down into several key steps:

i **Workflow Modeling:** Define the Workflow
Identify individual tasks in the workflow and their dependencies. Represent the workflow as a Directed Acyclic Graph (DAG) or another suitable model.

ii **Resource Description:** Specify Resource Characteristics
Define the characteristics of available cloud resources, including virtual machines (VMs) or containers. Consider factors such as processing power, memory, storage, and network bandwidth.

iii **Task Mapping and Dependency Management:** Map Tasks to Resources
Assign tasks to appropriate cloud resources based on their characteristics. Consider task dependencies to ensure correct execution order.

iv **Scheduling Algorithm:** Select Scheduling Algorithm
Choose a scheduling algorithm based on the workflow characteristics and objectives. Common algorithms include First-Come-First-Serve (FCFS), Round Robin, and more advanced algorithms like HEFT (Heterogeneous Earliest Finish Time).

v **Resource Allocation Optimization:** Optimize Resource Allocation
Use scheduling algorithms to optimize the allocation of resources, considering factors such as cost, resource availability, and performance objectives. Minimize resource contention and balance the load across available resources.

vi **Feedback and Optimization:** Analyze Results
Analyze the results of the workflow execution. Identify areas for improvement and optimization based on performance metrics.

1.3 Challenges in Task Scheduling for Cloud Computing Environments

Task scheduling in cloud computing involves the allocation of computational resources to tasks in a manner that optimizes various performance metrics, such as completion time, resource utilization, and energy efficiency. The complexity of task scheduling arises from the dynamic and heterogeneous nature of cloud environments, where multiple tasks with diverse resource requirements compete for shared resources.

The task scheduling problem in cloud computing is inherently challenging due to the following factors:

i **Heterogeneity**: Cloud data centers typically comprise a diverse set of resources, including virtual machines (VMs) with different configurations, storage options, and network capabilities. Efficiently scheduling tasks on such heterogeneous resources requires sophisticated algorithms that consider these variations.

ii **Dynamic Workloads**: Cloud environments experience dynamic fluctuations in workload, with varying numbers of incoming tasks and changing resource demands. Task schedulers must adapt in real-time to ensure optimal resource allocation, taking into account the evolving nature of the workload.

iii **Multi-objective Optimization**: Task scheduling involves optimizing multiple conflicting objectives, such as minimizing task completion time, maximizing resource utilization, and reducing energy consumption. Balancing these objectives requires intelligent scheduling strategies that consider the trade-offs between different performance metrics.

iv **QoS Requirements:** Cloud applications often have specific Quality of Service (QoS) requirements, such as response time, throughput, and reliability. Task scheduling algorithms must consider these requirements to ensure that performance goals are met.

1.4 Heuristics and Meta heuristics Techniques for Optimization Problems

1.4.1 Heuristics Techniques

Heuristics are rule-of-thumb strategies or algorithms designed to quickly find good, though not necessarily optimal, solutions to problems. They leverage domain-specific knowledge and intuition to guide the search towards promising regions of the solution space. The two broadly classified Heuristic Algorithms are:

i. **Greedy Heuristics:** Greedy algorithms make locally optimal choices at each step, aiming for a globally optimal solution. They are computationally efficient but may not always yield the best solution.

ii. **Constructive Heuristics:** Constructive heuristics build a solution step by step. Starting with an empty solution, elements are added incrementally based on certain criteria until a complete solution is reached. The construction process is guided by heuristics to make efficient choices.

1.4.2 Meta Heuristics Techniques

Meta heuristics are higher-level strategies that guide the exploration of the solution space. They are more generic and adaptable, providing a framework for finding near-optimal solutions across a wide range of problems. Meta heuristics are often iterative and stochastic, incorporating randomness to escape local optima. The majorly used Meta Heuristics Algorithms are as follows:

i. **Genetic Algorithms (GAs):** Inspired by natural selection and genetics, GAs operate on a population of potential solutions. Through processes like crossover and

mutation, new solutions are generated, and the fittest individuals are selected for the next iteration.

ii. **Simulated Annealing:** Simulated annealing gradually reduces the temperature during the search, allowing the algorithm to explore a broad solution space at higher temperatures and converge towards a more focused search at lower temperatures.

iii. **Particle Swarm Optimization (PSO):** Modeled after the social behavior of birds or fish, PSO involves a population of particles moving through the solution space. Each particle adjusts its position based on its own experience and the collective knowledge of the swarm, seeking to converge toward an optimal solution.

iv. **Ant Colony Optimization (ACO):** Inspired by the foraging behavior of ants, ACO constructs solutions incrementally. Artificial ants deposit pheromones on the paths they traverse, and the intensity of pheromones influences the likelihood of subsequent ants choosing the same paths.

1.4.3 Advantages of Heuristics and Meta Heuristics

i. **Applicability:** Heuristics and meta heuristics can be applied to a wide range of optimization problems without requiring problem-specific modifications.

ii. **Efficiency:** These methods often outperform exact algorithms in terms of computational efficiency, making them suitable for large and complex problem instances.

iii. **Flexibility:** Heuristics and meta heuristics can adapt to various problem structures and types, making them versatile in addressing different optimization challenges.

iv. **Global Exploration:** Meta heuristics, in particular, excel at global exploration of the solution space, helping to avoid getting stuck in local optima.

Difference between Heuristics and Meta Heuristics have been shown in the Table 1.1.

1.5 Simulation Tool: CloudSim

CloudSim, as a simulation tool, plays a pivotal role in advancing research in cloud computing by providing a comprehensive environment for modeling and analyzing cloud infrastructure and services. Developed by Calheiros et al. in 2011 [16], CloudSim has become a fundamental resource for researchers, offering a level of abstraction that enables them to concentrate on their specific research objectives without delving into the intricate details of cloud infrastructure and services.

Characteristic	**Heuristic Techniques**	**Meta-Heuristic Techniques**
Dependency	Problem-dependent techniques.	Problem-independent techniques.
Adaptation	Tailored to the specific problem's characteristics.	Designed to be adaptable to a wide range of problems.
Nature	Greedy approach, making locally optimal decisions.	Exploration-oriented, balancing exploration over exploitation.
Risk of Local Optima	May get trapped in local optima, potential for sub optimal solutions.	Less prone to local optima due to exploration and broad search.

Table 1.1 Difference between heuristics and meta heuristics techniques

The key components provided by CloudSim include Data Centers, Hosts, Virtual Machines (VMs), Cloudlets, and Data Center Broker. These components collectively simulate the essential elements of a cloud environment, allowing researchers to design, implement, and evaluate various algorithms and strategies for resource management, task scheduling, and overall system optimization.

Within CloudSim's framework, Data Centers serve as the core infrastructure, housing Hosts that, in turn, accommodate Virtual Machines[15]. This hierarchical structure mirrors the actual architecture of cloud computing environments. The scheduling of VMs is a critical aspect of cloud management, as it directly impacts resource utilization, performance, and energy efficiency. CloudSim enables researchers to focus on the intricacies of VM scheduling algorithms without the need to grapple with the complexities of real-world cloud infrastructure.

Furthermore, the presence of the Data Center Broker in CloudSim is significant. Acting as an intermediary between the user and the cloud infrastructure, the Data Center Broker abstracts the management of VMs, allowing researchers to design and evaluate scheduling policies from the user's perspective. This abstraction is crucial for simulating realistic cloud scenarios where users interact with cloud services through brokers, and it simplifies the investigation of user-centric metrics such as response time, throughput, and cost.

CloudSim serves as a valuable tool for researchers in the field of cloud computing, offering a simulated environment that mimics the essential components of a cloud system. By providing this abstraction, CloudSim facilitates the exploration and refinement of algorithms and strategies related to VM scheduling, resource management, and overall

system performance[26]. As a result, researchers can gain insights into the intricacies of cloud computing without the need to navigate the complexities of real-world cloud infrastructures.

The provided steps outline the process of setting up and executing a simulation in CloudSim, specifically for cloud computing scenarios involving data centers, hosts, virtual machines (VMs), and cloudlets. Here's an explanation of each step as shown in Fig. 1.8.

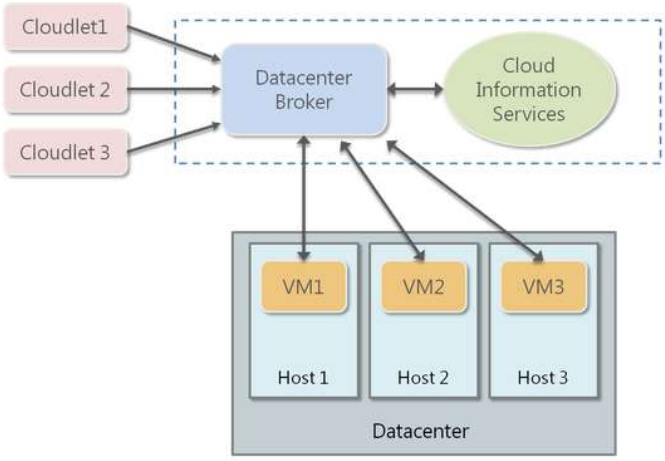

Fig. 1.8 Structure of CloudSim simulation tool

i **Create DataCenter in CloudSim:**
Configure the DataCenter entity in CloudSim, specifying characteristics such as architecture, OS, VM scheduling policy, and other relevant parameters. Include multiple hosts within the DataCenter to model the physical machines that will execute VMs.

ii **Create DataCenter Broker:**
Instantiate a DataCenter Broker, which acts as an intermediary between users (or tasks) and the cloud infrastructure. Configure scheduling policies and algorithms within the broker to optimize task allocation.

iii **Create Virtual Machines and Add to VMList:**
Create Virtual Machine entities with specific characteristics (e.g., MIPS, RAM, storage) and add them to the VMList. Associate the VMList with the DataCenter Broker, allowing the broker to allocate appropriate hosts for each VM.

iv **Create Cloudlet (Task) and Bind to Broker:**
Create Cloudlet entities representing user tasks, specifying parameters such as length, input/output file sizes, and resource requirements. Bind the Cloudlet entities to the DataCenter Broker, indicating that the broker is responsible for scheduling and executing these tasks.

v **Start the Simulation:**
Start the CloudSim simulation engine. The simulation runs, and tasks are scheduled based on the defined algorithms, with VMs executing cloudlets on allocated hosts.

vi **Obtain Results:**
Monitor and record various simulation metrics, such as task completion times, resource utilization, and overall system efficiency. Use the obtained results to assess the effectiveness of the scheduling algorithm and the overall cloud infrastructure configuration.